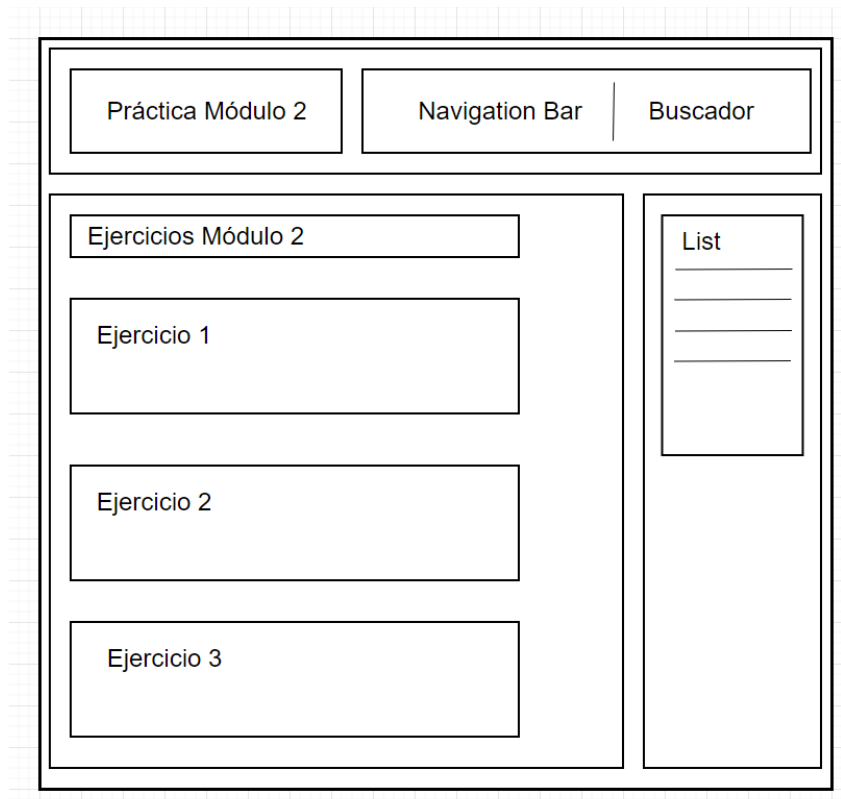
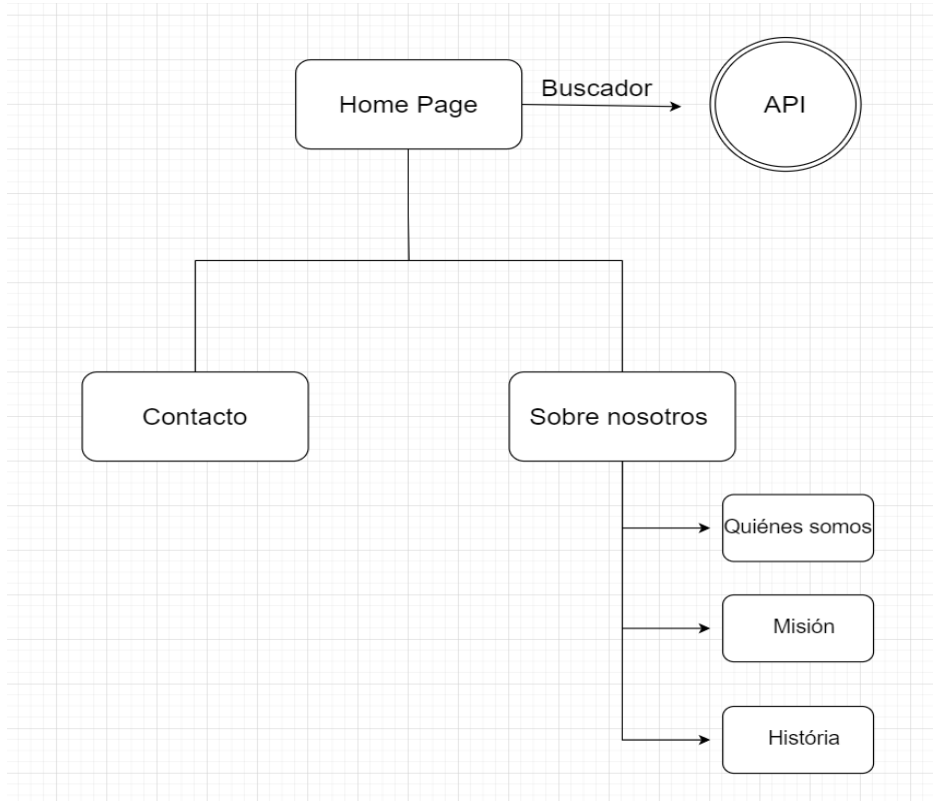


PRÁCTICA MÓDULO 2

1. Crea los storyboards o guión gráfico de cada una de las páginas. Cuestiones a considerar: Otras páginas relacionadas. Tipos de usuarios. Árbol web. Plantillas HTML5 o de Bootstrap. Mobile first. Vista móvil, tablety computador.



2. Asocia los diferentes elementos de las páginas del Storyboard diseñadas con cada elemento semántico de HTML5.

3. Aplicar los estilos CSS según los ejercicios propuestos en el módulo 2.2

Práctica Módulo 2

[Home](#) [Contacto](#) [Sobre nosotros](#)

Ejercicios Módulo 2

Ejercicio 4

Escribe un programa que declare 3 objetos de cada modelo de datos considerado: gestor, cliente, mensaje y transferencia. Los valores de las propiedades de los objetos pueden ser arbitrarios.

Dale al botón para mostrar resultados.

Mostrar Ejercicio 4

Ejercicio 5

Escribe un programa que almacene los objetos creados en el ejercicio anterior del proyecto dentro de un array (un array por cada modelo de datos). A continuación, recorre cada uno de los arrays y muestra todas propiedades.

Dale al botón para mostrar resultados.

Mostrar ejercicio 5

Ejercicio 6

Escribe un programa que realice la conversión a JSON del array (y viceversa) creado en el ejercicio anterior del proyecto.

Dale al botón para mostrar resultados.

Mostrar Ejercicio 6 (Array to JSON)

Mostrar Ejercicio 6 (JSON to Array)

A list item

14

A second list item

2

A third list item

1

A fourth list item

16

A fifth list item

21

A sixth list item

1

A seventh list item

3

Escribe aquí para buscar

Resumen Práctica ...

GitHub Desktop

Fox - Adobe Acrob...

Práctica Módulo 2 ...

main.js - Práctica ...

Práctica Módulo 2...

23°C

17:44

11/10/2022

4. Escribe un programa que declare 3 objetos de cada modelo de datos considerado: gestor, cliente, mensaje y transferencia. Los valores de las propiedades de los objetos pueden ser arbitrarios.

```
class modeloDeDatosBanco {
  constructor(gestor, cliente, mensaje, transferencia) {
    this.gestor = gestor;
    this.cliente = cliente;
    this.mensaje = mensaje;
    this.transferencia = transferencia;
  }
}

let Object1 = new modeloDeDatosBanco("Manuel", "Sergio", "Nómina Agosto", 1000)
let Object2 = new modeloDeDatosBanco("Fernando", "Silvia", "Gastos de transporte", 200)
let Object3 = new modeloDeDatosBanco("Roberto", "Adrián", "Finiquito", 800)

btnEjercicio4.addEventListener("click", () =>{
  console.warn("Ejercicio 4")
  console.log(Object1);
  console.log(Object2);
  console.log(Object3);
  alert("Mira el resultado por consola!")
})
```

5. Escribe un programa que almacene los objetos creados en el ejercicio anterior del proyecto dentro de un array (un array por cada modelo de datos). A continuación, recorre cada uno de los arrays y muestra todas las propiedades.

```
let arrayObject1 = [];  
let arrayObject2 = [];  
let arrayObject3 = [];  
  
const objectsToArray = (object, arrayObject) =>{  
    Object.entries(object).forEach(element => {  
        arrayObject.push(element)  
    });  
    console.log("Array del objeto: ", arrayObject);  
}  
  
btnEjercicio5.addEventListener("click", () =>{  
    console.warn("Ejercicio 5")  
    objectsToArray(Object1, arrayObject1)  
    objectsToArray(Object2, arrayObject2)  
    objectsToArray(Object3, arrayObject3)  
  
    alert("Mira el resultado por consola!")  
})
```

6. Escribe un programa que realice la conversión a JSON del array (y viceversa) creado en el ejercicio anterior del proyecto.

```
let json1 = ''
let json2 = ''
let json3 = ''

let conversorArrayToJson = (array) => {

  let json = JSON.stringify(array)
  console.log("Json del Array: ", json);
  return json;
}

let conversorJsonToArray = (json) => {

  let array = JSON.parse(json)
  console.log("Array del Json: ", array);
}

btnEjercicio6a.addEventListener("click", () => {

  if(arrayObject1.length === 0){
    alert("Haz primero el ejercicio 5 para continuar")
  }else{
    alert("Mira el resultado por consola!")
    console.warn("Ejercicio 6 (Array to JSON):")
    json1 = conversorArrayToJson(arrayObject1)
    json2 = conversorArrayToJson(arrayObject2)
    json3 = conversorArrayToJson(arrayObject3)
  }
});

btnEjercicio6b.addEventListener("click", () => {

  if(json1 === ''){
    alert("Haz primero el ejercicio 6 (Array to Json) para continuar")
  }else{
    alert("Mira el resultado por consola!")
    console.warn("Ejercicio 6 (JSON to Array):")
    conversorJsonToArray(json1)
    conversorJsonToArray(json2)
    conversorJsonToArray(json3)
  }
});
```

7. Escribe un programa que haga uso del servicio web del banco y realice una petición con AJAX a la url: <http://localhost:8085/ok>

-Primero he creado un servidor local con NodeJs y express, que una vez corre con “npm start”, ante las peticiones a <http://localhost:8085/ok>, devuelve los modelos de datos del banco en formato Json. Para que funcione hay que instalar NodeJs y Cors.

-Después, en mi archivo principal main.js, que ejecuta mi web en el navegador, mediante un botón hago la petición al servidor, que imprime por pantalla el resultado en Json.

```
JS server.js X
JS server.js > json > "banco"
1
2 const express = require('express')
3 let cors = require("cors");
4 const app = express()
5
6
7 app.use(cors());
8 const port = 8085
9
10
11
12
13 let json = {
14   "banco": [
15     {
16       "gestor": "Manuel",
17       "cliente": "Sergio",
18       "mensaje": "Nómina Agosto",
19       "transferencia": 1000
20     },
21     {
22       "gestor": "Fernando",
23       "cliente": "Silvia",
24       "mensaje": "Gastos de transporte",
25       "transferencia": 200
26     },
27     {
28       "gestor": "Roberto",
29       "cliente": "Adrián",
30       "mensaje": "Finiquito",
31       "transferencia": 800
32     }
33   ]
34 }
35
36 app.get('/ok', (req, res) => {
37   res.send(json)
38 })
39
40 app.listen(port, () => {
41   console.log(`Example app listening on port ${port}`)
42 })
```

```
const peticionAjax = async () => {
  let request = await $.ajax({
    method: "GET",
    headers: { 'Access-Control-Allow-Origin': 'http://localhost:8085/ok' },
    url: "http://localhost:8085/ok"
  });

  return request;
}

btnEjercicio7.addEventListener("click", async () => {

  console.warn("Ejercicio 7")

  let respuesta = await peticionAjax()

  console.log(respuesta);

  alert("Mira el resultado por consola")
});
```

8. Crea una función en JavaScript que obtenga todos los gestores de forma periódica cada 5 segundos.

```
let interval;
btnEjercicio8.addEventListener("click", () => {

    console.warn("Ejercicio 8")
    console.log("Inicio de interval");
    interval = setInterval(async() => {

        let data = await peticionAjax()

        if (data){
            data.banco.forEach(gestion => {
                gestor = gestion.gestor
                console.log({gestor});
            })
        }

    }, 5000);

});

btnEjercicio8stop.addEventListener("click", () => {

    clearInterval(interval)
    console.log("Interval parado");
});
```

9. Migra el código a una aplicación TypeScript.

-He creado un archivo nuevo llamado "ejercicio9-JstoTS.ts", que contiene el mismo código que main.js pero en versión ampliada a TypeScript.

-Ejecutando el comando "tsc ejercicio9-JstoTS.ts", transpila el código a Javascript creando un nuevo archivo llamado "ejercicio9-JstoTS.js"

EJEMPLO DE UN TROZO DEL CÓDIGO:

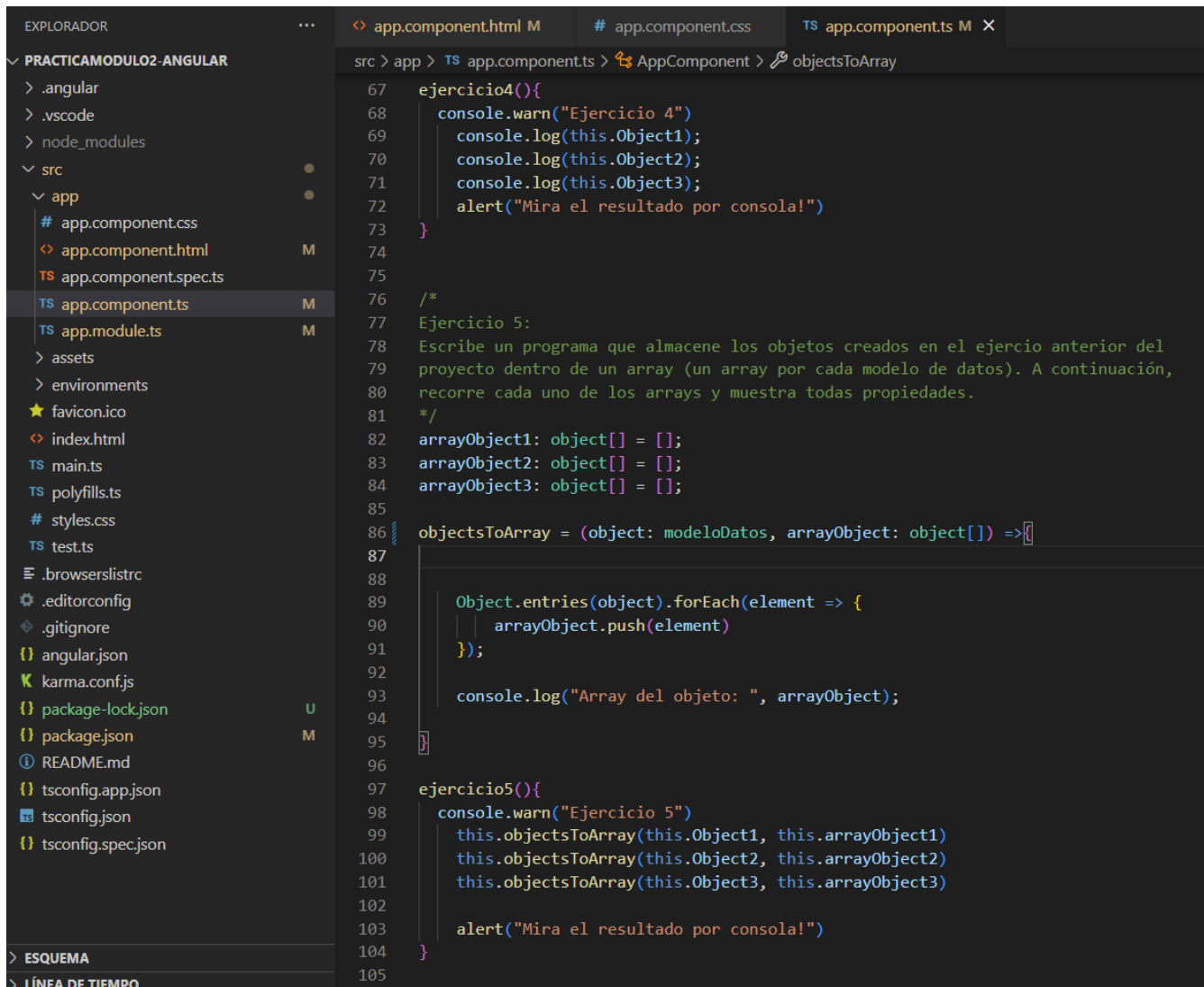
```
TS ejercicio9-JstoTS.ts M X
TS ejercicio9-JstoTS.ts > [?] conversorArrayToJson
19  pueden ser arbitrarios.
20  */
21
22  interface modeloDatos {
23      gestor: string
24      cliente: string
25      mensaje: string
26      transferencia: number
27  }
28
29
30  class modeloDeDatosBanco {
31      gestor: string
32      cliente: string
33      mensaje: string
34      transferencia: number
35
36      constructor(gestor: string, cliente: string, mensaje: string, transferencia: number) {
37          this.gestor = gestor;
38          this.cliente = cliente;
39          this.mensaje = mensaje;
40          this.transferencia = transferencia;
41      }
42  }
43
44
45  let Object1: modeloDatos = new modeloDeDatosBanco("Manuel", "Sergio", "Nómina Agosto", 1000)
46  let Object2: modeloDatos = new modeloDeDatosBanco("Fernando", "Silvia", "Gastos de transporte", 200)
47  let Object3: modeloDatos = new modeloDeDatosBanco("Roberto", "Adrián", "Finiquito", 800)
48
```

10. Migra el código a una aplicación Angular.

He creado un nuevo proyecto Angular, que se puede encontrar en la carpeta “Práctica 2 -Ejercicio 10 (en Angular)”.

He adaptado algunas partes del código como los “getElementById” y “addEventListener”, haciendolo más simple con herramientas de Angular.

También he tenido que re-adaptar la petición ajax a una petición con HttpClient.



The screenshot shows the Visual Studio Code interface with an Angular project. The Explorer on the left shows the project structure, including the 'src/app' directory. The main editor displays the 'app.component.ts' file, which contains two exercises. Exercise 4 is a simple function that logs and alerts object properties. Exercise 5 is a function that takes an object and an array, iterates over the object's entries, and pushes them into the array. The code is as follows:

```
67 ejercicio4(){
68   console.warn("Ejercicio 4")
69   console.log(this.Object1);
70   console.log(this.Object2);
71   console.log(this.Object3);
72   alert("Mira el resultado por consola!")
73 }
74
75
76 /*
77 Ejercicio 5:
78 Escribe un programa que almacene los objetos creados en el ejercicio anterior del
79 proyecto dentro de un array (un array por cada modelo de datos). A continuación,
80 recorre cada uno de los arrays y muestra todas propiedades.
81 */
82 arrayObject1: object[] = [];
83 arrayObject2: object[] = [];
84 arrayObject3: object[] = [];
85
86 objectsToArray = (object: modeloDatos, arrayObject: object[]) => {
87
88   Object.entries(object).forEach(element => {
89     arrayObject.push(element)
90   });
91
92   console.log("Array del objeto: ", arrayObject);
93
94 }
95
96
97 ejercicio5(){
98   console.warn("Ejercicio 5")
99   this.objectsToArray(this.Object1, this.arrayObject1)
100   this.objectsToArray(this.Object2, this.arrayObject2)
101   this.objectsToArray(this.Object3, this.arrayObject3)
102
103   alert("Mira el resultado por consola!")
104 }
105
```